Amendments to the Claims:

Rewrite the claims as set forth below. This listing of claims replaces all prior versions and listings of claims in the application:

1. (currently amended) A multi-thread command processing system comprising:

a reservation station having a plurality of command threads stored therein;

an arbiter operably coupled to the reservation station such that the arbiter retrieves a first command thread of the plurality of command threads stored therein; and

a command processing engine operably coupled to receive the first command thread from the arbiter such that the command ~~processor~~ processing engine performs at least one processing command from the first command thread and thereupon updates the first command thread in the reservation station.

2. (currently amended) The command processing system of claim 1 wherein ~~each~~ a done flag is set in one of the plurality of command threads ~~include a done flag which is activated~~ after all commands ~~with~~ within the one of the plurality of command ~~thread~~ threads have been executed by the command processing engine.

3. (original) The command processing system of claim 1 wherein the reservation station is a memory device and the arbiter retrieves the first command thread based on a priority scheme.

4. (currently amended) The command processing system of claim 1 ~~further comprising~~ wherein:

the arbiter retrieves a second command thread of the plurality of command threads stored therein; and

the command processing ~~receiving~~ receives the second command thread wherein the first command thread and the second command thread may be interleaved.

5. (original) The command processing system of claim 1 wherein the command processing engine is a texture processing engine.

6. (original) The command processing system of claim 1 further comprising:

an arithmetic logic unit (ALU) operably coupled to the arbiter such that the arbiter is capable of providing at least one of the plurality of command threads to the ALU.

7. (original) The command processing system of claim 1 wherein the reservation station is at least one of: a vertex reservation station and a pixel reservation station.

8. (currently amended) A multi-thread graphics processing system comprising:

a first reservation station having a plurality of first command threads stored therein;

a second reservation station having a plurality of graphic command threads stored therein;

an arbiter coupled to the first reservation station and the second reservation station such that the arbiter retrieves a selected command thread from one of the plurality of first command threads and the plurality of graphic command threads; and

a graphics processing engine operably coupled to the arbiter such that the selected command thread is received and processed by the graphics processing engine.

9. (currently amended) The multi-thread graphics processing system of claim 8 ~~further comprising: the~~ wherein the graphics processing engine ~~being~~ is operably coupled to the first reservation station and the second reservation station such that

upon processing the selected command thread, the selected command thread is updated in the first reservation station ~~or the second reservation station from wherein is was previously retrieved~~ if the selected command thread was previously retrieved from the first reservation station, and the selected command thread is updated in the second reservation station if the selected command thread was previously retrieved from the second reservation station.

10. (original) The multi-thread graphics processing system of claim 8 wherein the first reservation station is a memory device and the second reservation station is a FIFO memory device.

11. (currently amended) The multi-thread graphics processing system of claim 8 wherein ~~further comprising~~:

the arbiter retrieves a second selected command thread from one of the plurality of command threads; and

the command processing engine ~~receiving~~ receives the second selected command thread such that the first selected command thread and the second selected command thread may be interleaved.

12. (original) The multi-thread graphics processing system of claim 8 wherein the graphics processing engine is a texture processing engine, the system further comprising:

an arithmetic logic unit (ALU) operably coupled to the arbiter such that the arbiter is capable of providing at least one of the first selected command thread and the second selected command thread to the ALU.

13. (original) The multi-thread graphics processing system of claim 8 wherein the first reservation station is a vertex reservation station and the second reservation station is a pixel reservation station.

14. (currently amended) A graphics processing system comprising:

a pixel reservation station having a plurality of pixel command threads stored therein;

a ~~vector~~ vertex reservation station having a plurality of ~~vector~~ vertex command threads stored therein;

an arbiter coupled to the pixel reservation station and the ~~vector~~ vertex reservation station;

an arithmetic logic unit (ALU) operably coupled to the arbiter; and

a texture engine operably coupled to the arbiter wherein the arbiter retrieves a first selected command thread from one of the plurality of pixel command threads and the plurality of ~~vector~~ vertex command threads and the arbiter thereupon provides the first selected command thread to at least one of: the ALU and the texture engine.

15. (currently amended) The graphics processing system of claim 14 ~~further comprising:~~

~~the~~ wherein the arbiter retrieves a second selected command thread from one of the plurality of pixel command threads and the plurality of ~~vector~~ vertex command threads and thereupon provides the second selected command thread to at least one of the ALU and the texture engine.

16. (currently amended) The graphics processing system of claim 15 wherein when the first selected command thread and the second selected command thread are both provided to the ALU, such that the first selected command thread may be interleaved with the second selected command thread.

17. (currently amended) The graphics processing system of claim 14 wherein the ALU and the texture engine are operably coupled to the pixel reservation station and the ~~vector~~ vertex reservation station such that the first selected command thread may be provided back ~~from wherein it~~ to the pixel reservation station if the first selected command thread was previously retrieved from the pixel reservation station, and such that the first selected command thread may be provided back to the vertex reservation station if the first selected command thread was previously retrieved from the vertex reservation station.

18. (currently amended) The graphics processing system of claim 14 wherein the pixel reservation station includes a first pixel memory device storing a plurality of pixel state bits and a second pixel memory device storing a plurality of pixel status bits and the ~~vector~~ vertex reservation station includes a first ~~vector~~ vertex memory device storing a plurality of ~~vector~~ vertex state bits and a second ~~vector~~ vertex memory device storing a plurality of ~~vector~~ vertex status bits.

19. (currently amended) The graphics processing system of claim 14 wherein each of the plurality of pixel command threads and each of the ~~vector~~ plurality of vertex command threads ~~each~~ include a done flag, wherein when the done flag is ~~activated~~ set in one of the plurality of pixel command ~~thread~~ threads, the one of the plurality of pixel command ~~thread~~ threads is provided to a rendering backend and when the done flag is ~~activated~~ set in one of the

~~vector~~ plurality of vertex command ~~thread~~ threads, the ~~vector~~ one of the plurality of vertex command ~~thread~~ threads is provided to a scan converter.

20. (currently amended) A method for multi-thread graphics processing comprising:

retrieving a selected command thread from a plurality of command threads;

providing the selected command thread to a graphics processing engine;

performing a command in response to the selected command thread; and

writing the selected command thread to a first reservation station if the selected command thread is one of a plurality of first command threads and writing the selected command thread to a second reservation station if the selected command thread is one of a plurality of second command threads.

21. (currently amended) The method of claim 20 further comprising:

retrieving a second selected command thread from the plurality of command threads;

providing the second selected command thread to at least one of the graphics processing engine and an arithmetic logic unit (ALU);

prior to writing the selected command thread to ~~either~~ one of the first station ~~or~~ and the second reservation station, interleaving the selected command thread and the second selected command thread;

performing a second command in response to the second selected command thread; and

writing the second selected command thread to a first reservation station if the second selected command thread is one of the plurality of first command threads and writing the second selected command thread to a second reservation station if the second selected command thread is one of the plurality of second command threads.

22. (cancelled)

23. (original) The method of claim 20 wherein the graphics processing engine is a texture engine.

24. (currently amended) The method of claim 20 ~~further comprising:~~

~~when~~ wherein when all of the commands with the selected command thread have been executed, setting a done flag.

25. (currently amended) The method of claim 24 wherein the ~~first~~ plurality of first command threads ~~are~~ comprises a plurality of pixel command threads and wherein the ~~second~~ plurality of second command threads ~~are~~ comprises a plurality of vertex command threads, the method further comprising:

when the done flag is set ~~on~~ in one of the plurality of pixel command ~~thread~~ threads, providing the one of the plurality of pixel command ~~thread~~ threads to a render backend; and

when the done flag is set ~~on~~ in one of the plurality of vertex command ~~thread~~ threads, providing the one of the plurality of vertex command ~~thread~~ threads to a scan converter.